# Functions and scope

# Functions

```
fn main() {
   say_hi();
}

fn say_hi() {
   println!("Hello there!");
}
```

# Functions

```
fn main() {
   say_hi("John");
}

fn say_hi(name: &str) {
   println!("Hello {}!", name);
}
```

# Functions

```
fn main() {
   let mut name = "John";
   say_hi(&mut name);
   print!("The new name is {}", name)
}

fn say_hi(name: &mut &str) {
   *name = "Alex";
   println!("Hello {}!", name);
}
```

# Functions

```
fn main() {
   let mut name = "John";
   let greeting = say_hello(&mut name);
   println!("{}", greeting);
}

fn say_hello(name: &mut &str) -> String {
   let greeting = format!("Hello {}", name);
   greeting
}
```

# Scope

No memory leaks - no need to manually deallocate variables

# Scope

No memory leaks - no need to manually deallocate variables

```
{
   let a = 3;
}
println(“a = {}”, a)              // error
```

# Scope

No memory leaks - no need to manually deallocate variables

```
{
  let a = 3;
}
println(“a = {}”, a)              // error
```

Global variables can be declared but they are unsafe

```
let a = 3;
```

# Scope

## No memory leaks - no need to manually deallocate variables

```
{
  let a = 3;
}
println(“a = {}”, a)              // error
```

## Global variables can be declared but they are unsafe

```
let a = 3;
```

```
fn main() {
  unsafe { println!(“{}”, a); }
}
```